John von Neumann Institute for Computing

A Parallel Adaptive Algorithm to Improve Precision of Time Series Identification

J.A. Gómez, M.A. Vega, J.M. Sánchez, J.M. Granado

published in

Parallel Computing:
Current & Future Issues of High-End Computing,
Proceedings of the International Conference ParCo 2005,
G.R. Joubert, W.E. Nagel, F.J. Peters, O. Plata, P. Tirado, E. Zapata (Editors),
John von Neumann Institute for Computing, Jülich,
NIC Series, Vol. **33**, ISBN 3-00-017352-8, pp. 293-300, 2006.

http://www.fz-juelich.de/nic-series/volume33

A Parallel Adaptive Algorithm to Improve Precision of Time Series Identification

Juan Antonio Gomez Pulido[a], Miguel Angel Vega Rodriguez[a], Juan Manuel Sanchez Perez[a], Jose Maria Granado Criado[a]

[a]Escuela Politecnica, University of Extremadura, 10071 Caceres, Spain

1. Introduction

In this work we present a parallel technique to optimize time series modelling in order to obtain high precision predictions. Also, this technique could be very useful when the high precision mathematical modelling of dynamic complex systems is required. We employ System Identification algorithms, and use recursive least squares processing and ARMAX modelling. After explaining the proposed heuristic (a set of parallel processing units that performs an adaptive algorithm) and the tuning of its parameters, we show the results we have found for several benchmarks. Thus, we demonstrate how the result precision improves.

2. Modeling and Predicting Time Series

In many engineering fields it is necessary to dispose of mathematical models for studying the behaviour of dynamic systems whose mathematical description is not available "a priori". One interesting type of these systems is the Time Series (TS). Time series are used to describe systems in many fields: meteorology, economy, physics, etc. When dealing with TS there is only available a signal under observation; its physical structure is not known. This led us to employ planning System Identification (SI) techniques [1] in order to obtain the TS model. The model precision depends on the assigned values to certain parameters. In this paper we propose a heuristic to adjust these parameters with the aim of improving the precision of the model.

We consider a TS as the description of a simple dynamic system, by means of a sampled signal with period T that is modelled with an ARMAX [2] parametric description (see equation 1: ARMAX model of a TS, where na is the model dimension).

$$y(k) + a_1 y(k_1) + ... + a_{na} y(k_{na}) = 0 \tag{1}$$

Basically the identification consists in determining the ARMAX model parameters a_i (θ in matricial notation) from measured samples $y(k_i)$ ($\varphi(k)$ in matricial notation). Then it is possible to compute the estimated signal $y_e(k)$ (equation 2: estimated value of the TS at k time) and compare it with the real signal y(k), computing the generated error at k time.

$$y_e(k) = [-a_1 y(k-1) - ... - a_{na} y(k-na)] = \varphi^T(k)\theta \tag{2}$$

The recursive estimation updates a_i in each time step k, thus modelling the system (fig. 1). The more sampled data processed, the more precision for the model, because it has more information about the system behaviour history. We consider SI performed by the Recursive Least Squares (RLS) with forgetting factor (λ) algorithm [2]. From the initial conditions, we build $\varphi^T(k)$, and then RLS runs iterations to evaluate y_e. This algorithm is specified by the constant λ, the initial values and the observed samples y(k). There is not any fixed value for λ, even it is used a value between

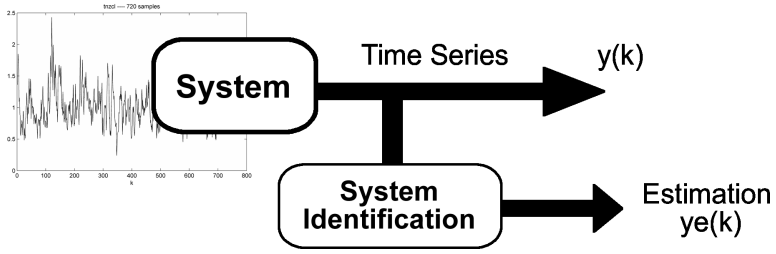

Figure 1. Time series identification

0.97 and 0.995 [3]. The cost function F (see equation 3, where SN is the sample number) is defined as the value to minimize in order to obtain the best precision.

$$F(\lambda) = \sum_{k=k_0}^{k=k_0+SN-1} |y_e(k) - y(k)| \tag{3}$$

The recursive identification is useful for predicting the system behaviour. For example, it could be interesting to know the future behaviour that cannot be experimentally predicted in the prevision of critical or emergency situations. However, for controlling purposes, it is necessary to make a prediction, and for predicting it is necessary to obtain information about the system. This information, acquired by means of the System Identification, consists in elaborating a mathematical model for covering the system behaviour under any working conditions, even under the more extremes.

SI allows finding, in sample time, a mathematical model from which is possible to predict future behaviours. As identification advances in the time, the predictions improve using more precise models. For example, we can compute in sample time the system model and then, with this model to simulate the system future behaviour, forwarding real situations.

3. Parameter Optimization to Improve Prediction Precision

When SI techniques are used, the model is generated "a posteriori" by means of the measured data. However we are interested in the system behaviour prediction in running time, that is, while the system is working and its data are being observed. So, it would be interesting to generate models in running time in such a way that a processor may simulate the system next behaviour.

At the same time, our first effort is to obtain a high model precision (minimal F). System Identification precision is due to several causes, mainly to the forgetting factor λ (fig. 2). Frequently this value is critical for model precision. Other sources also can have less degree of influence (model dimensions, etc), but they are considered as problem definitions, not parameters to be optimized.

On the other hand, it may appear the precision problem when a system model is generated in sample time: If the system response changes quickly, then the sample frequency must be high for avoiding the key data loss in the system behaviour description. If the system is complex and its simulation from the model to be found must be very trustworthy, then the required precision must be very high and this implies a great computational cost. Sometimes the hardware resources do not allow the computational cost in the model generation and processing to be lower than the sample period. We find the trade-off between a high sample frequency and a high precision in the algorithm computation. Adjusting the parameter λ for improving the precision can be conveniently done by means of the parallel technique we present in this work.

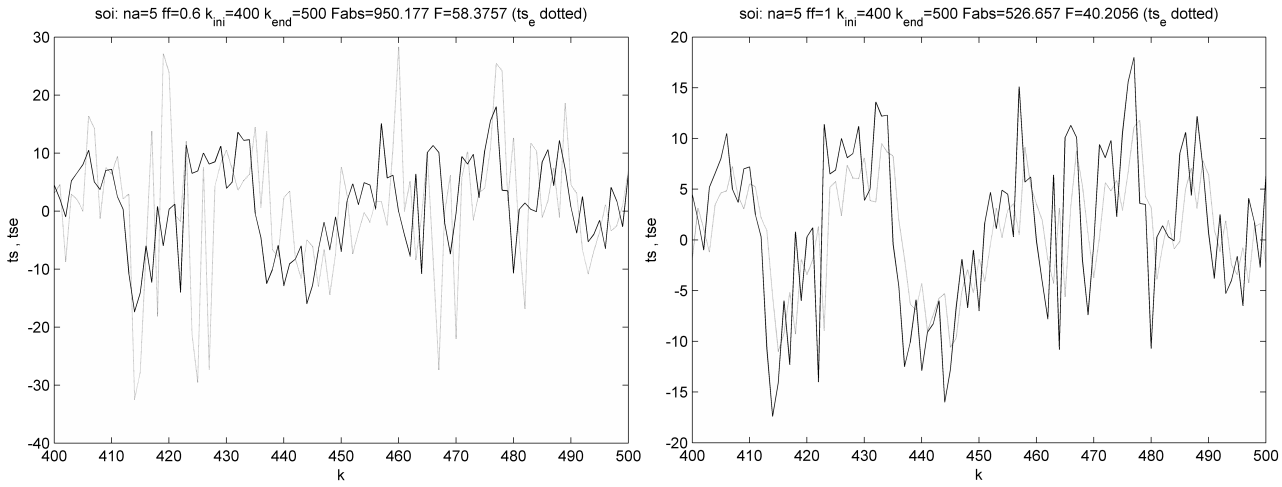

Figure 2. System Identification RLS of benchmark ball with different values for λ parameter. In the left case (λ=0.6) the produced error is greater than in the right case (λ=1)

A consideration prior to the application of the parallel algorithm for optimizing λ is setting the most adequate size of the models to generate in the identification. In other words: the degree of the polynomial expression ARMAX that represents the mathematical formulation of the time series. The notation of this parameter, which from now on we will call dimension of the model, is na.

In the fig. 3 the found results in the analysis performed for several time series benchmarks are shown. These benchmarks have been taken from some time series collections [4][5][6]. For each of them the following experimental procedure has been carried out:

- The results of the cost function F have been evaluated for 200 values of λ comprised to regular intervals in the range (0.9-1.1).
- The best of these obtained results is named Fopt, and written down for the value of the considered dimension of the model.
- This analysis is iterated for 147 different values of na, comprised in the range (3-150). The generated set of Fopt is graphically represented in the fig. 3.

It is deduced from the analysis of the behaviour of the fig. 3 that it can not talk about a general guideline that allows us to determine exactly which is the best dimension for the models. However, we can see the results worsen if the size of the model increases too much. From the data we have obtained here, together with the conclusions extracted from the experimentation carried out in other works [7], we can fix an adequate value of the models size in 15.

In order to find techniques to improve the precision in the time series prediction finding the best λ value, we propose a parallel adaptive algorithm, which is explained in depth in the next section.

4. A Parallel Adaptive Heuristic

In order to find the optimum value of λ, we propose a parallel algorithm that is partially inspired on the concept of artificial evolution [8][9] and also in simulated annealing mechanism [10]. In our algorithm, named PARLS (Parallel Adaptative Recursive Least Squares), the optimization parameter

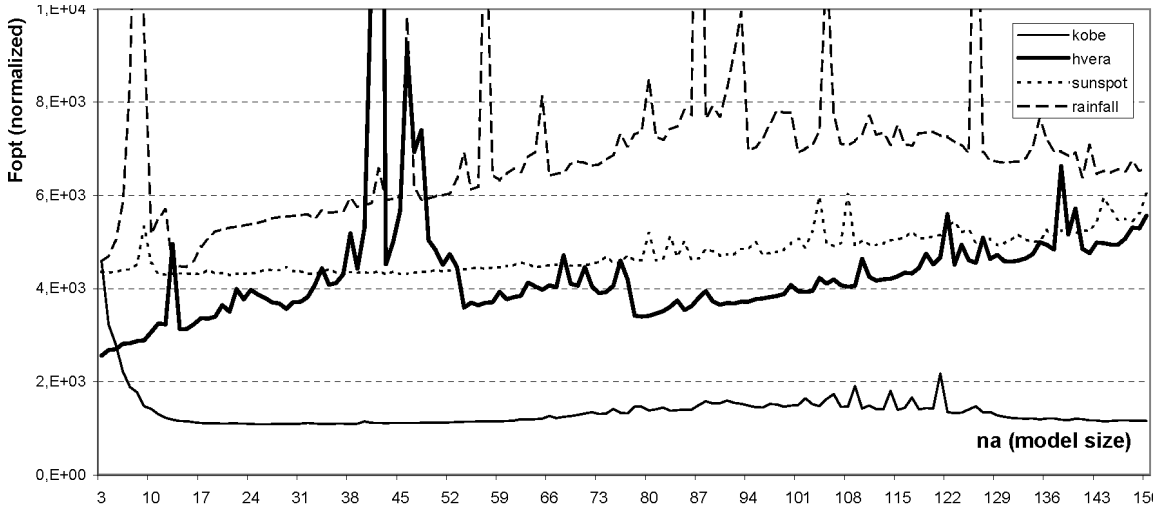

Figure 3. Smaller error (Fopt) in the estimation for 200 λ values vs. model size (ARMAX degree from 3 to 150). This analysis has been made for 4 time series benchmarks of different sizes

R	Generation interval
λc	λ central in R
PHS	Phase samples
PHN	Number of phases
PUN	Number of parallel processing units
TSN	Total number of samples
RED	Reduction factor of R

Table 1
PARLS nomenclature: the more important algorithm's parameters

λ is evolved for predicting new situations during the successive phases of the process. In other words, λ evolves at the same time that improves the cost function performance.

PARLS considers a λ value as a state. Starting on an initial λ value (λc) and an initial R value (the interval of generation where λc is in the middle), a set of λ values is generated covering uniformly the interval R. The λ values generated are equal to the number of parallel processing units (PUN). Each phase of PARLS is an identification loop that considers a given number of sample times (PHS) and the corresponding λ value. We use the nomenclature showed in Table 1.

In each phase, R is reduced dividing itself by the RED factor (the interval limits are moved so the center of interval corresponds with the optimal λ value found in the previous phase), in such a way that the generated set of λ will be more and more near of the previous optimum found. The new set of generated λ values covers uniformly the new R. In each processing unit, during each phase, the cost function F is computed (the accumulated error of the samples that constitutes each phase). From equation 3 we obtain the equation 4 as the cost function for each parallel processing unit.

$$F(\lambda_{PU_X}) = \sum_{k=k_0}^{k=k_0+PHS-1} |y_e(k) - y(k)| \tag{4}$$

At the end of each phase, the best λ is chosen. This is the corresponding value to the lower F. From this λ, new values are generated in a more reduced (new R) interval (see fig. 4). The goal

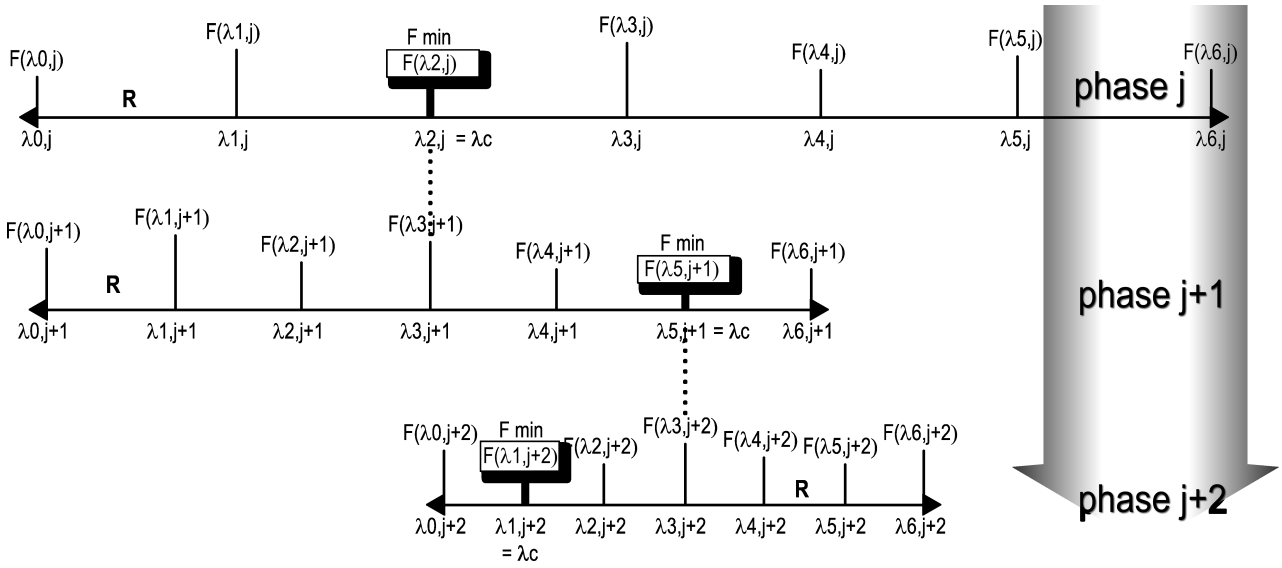

Figure 4. The SI uses different λ values in each phase performed by each processing unit. All the λ values in the same phase running in the processing units are generated in the R interval from the previous phase optimum λ found, corresponding with the smallest computed F

is that the identifications performed by the processing units will converge to optimum λ parameters when a given stop criteria will be achieved. In fig. 5 we can see this parallel architecture what would to help us to know better the PARLS performance.

5. Experimental Results

We consider several criteria for evaluating PARLS performances. All these criteria have been fully checked and tested in order to get a set of better values for parameters and strategies. For example, we have studied strategies as the optimum λ criteria (the λ value that produces a minimum F), the stop criteria (indicating when a processing unit must stop the work), the model generation criteria (how to consider the initial model in the next phase), the optimum F definition (to consider the optimum F as the lowest in all phases or the lowest computed in the present phase), etc.

Another important question is how to establish the initial range of λ values in PARLS search. We have performed several experiments in order to determine the approximated optimal λ for many benchmarks using a lot of RLS computations. In fig. 6 some of these experiments are shown. We can see that there is a distinct optimal λ for each benchmark, but in all cases there is a smooth U-curve that is very useful to the initial PARLS search. We have thus selected as initial searching parameters tuned values λc =1 and R=0.1.

PARLS offers a great variability for its parameters. We have carried out many experiments with a wide set of benchmarks. According to the results we have obtained, we can conclude that there are not common policies for tuning the parameters in such a way that always the best results will be found. But results indicate that there is a set of values for which the results are good. We can thus establish fixed values for PARLS parameters (see table 2) in order to define an unique algorithm applicable to any system.

In the table 3 we show the comparison of found results between RLS search and PARLS heuristic

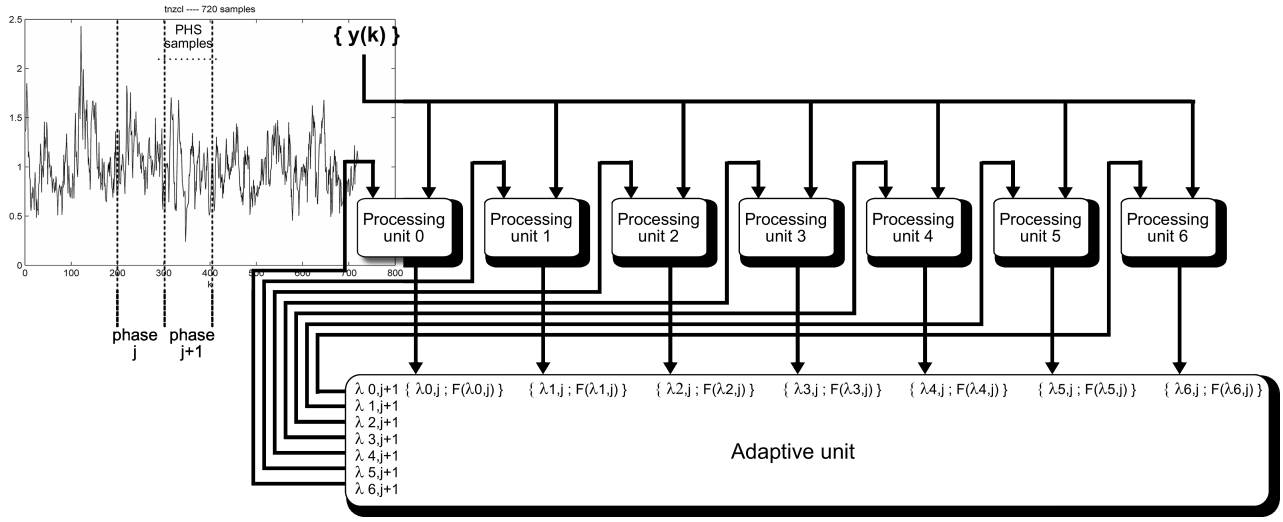

Figure 5. Parallel architecture for the adaptive high precision time series prediction

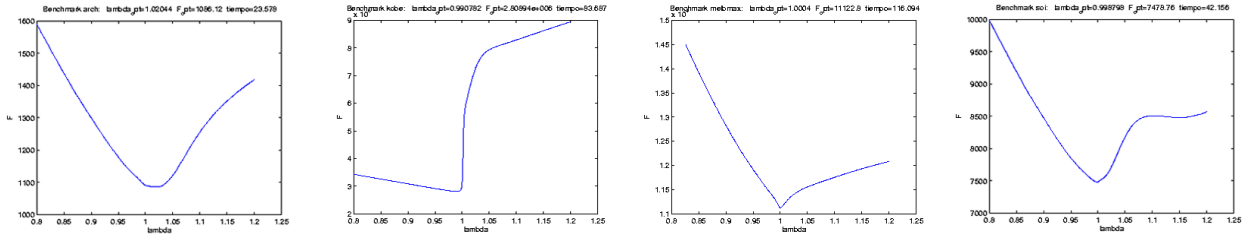

Figure 6. The cost function of six benchmarks calculated by RLS for 500 λ values in the same range (λc=1, R=0.4), using the dimension na=5. We can see a smooth U-curve in all the cases

Parameter	Tuned value
na	5
λc	1
R	0.05
PHN	4
PUN	11
RED	2

Table 2
Main PARLS parameter tuned

Benchmark	TSN	PHS	(a) F for RLS 11	(b) F for PARLS	Better algorithm	PARLS vs RLS 11
kobe	3.048	762	2.80896E+06	2.16053E+06	PARLS	30%
melbmin	3.648	912	7.28124E+03	7.46543E+03	RLS	-2%
melbmax	3.648	912	1.11282E+04	1.10042E+04	PARLS	1%
hvera	1.096	274	2.78909E+03	2.73085E+03	PARLS	2%
arch	1.000	250	1.08611E+03	1.03681E+03	PARLS	5%
f060	4.096	1.024	8.06796E+04	8.10842E+04	RLS	0%
n005	4.096	1.024	9.68213E+04	9.42284E+04	PARLS	3%
o094	4.096	1.024	4.01413E+04	3.87788E+04	PARLS	4%
s059	4.096	1.024	1.23377E+05	1.23183E+05	PARLS	0%
z002	4.096	1.024	3.65868E+04	3.70197E+04	RLS	-1%
soi	1.232	308	7.49888E+03	7.45276E+03	PARLS	1%
eeg01	18.432	4.608	4.24190E+05	6.70278E+04	PARLS	533%
eeg02	18.432	4.608	7.37492E+05	5.79659E+04	PARLS	1,172%
eeg03	15.360	3.840	4.42649E+04	4.29920E+04	PARLS	3%
eeg04	8.192	2.048	3.03256E+04	2.86598E+04	PARLS	6%
tnzcl	720	180	1.02752E+02	1.07755E+02	RLS	-5%
rainfall	3.652	913	7.54512E+04	6.93007E+04	PARLS	9%

Table 3
Comparison of results between RLS search and PARLS. The better algorithm is PARLS for the greater part of the benchmarks, and in the other cases the PARLS results is closely near to RLS search (see the % of PARLS better than RLS)

for several benchmarks. In all cases the same tuned parameters has been used (na=5, PUN=11, λc=1, R=0.05, RED=2, PHN=4). In (b) the results of 11 RLS identifications with their corresponding 11 equidistant in R values of λ are shown, and in (b) the PARLS results are displayed too. The computational effort of 11 RLS identifications is almost equal to PARLS cost with 11 processing units, so both results can be compared. With these tuned parameters, PARLS finds better results in the greater part of benchmarks. However, and this is important, for the other benchmarks, in all cases the difference of F oscillates between 2% and 5%. That is, PARLS improves or holds the results found with RLS with the same computational effort.

6. Conclusions and Future Works

With the tuned parameters and benchmarks considered, PARLS finds better results in the greater part of experiments. We can say the parallel adaptative heuristic PARLS offers a good performance, and this encourages us to follow this research. Also, a neural network implementation of the parallel processing units has been developed [11] in order to evaluate new computational costs, with good results. Now, our present effort is oriented to achieve a PARLS synthesis on reconfigurable hardware systems to improve the global efficiency [12], because it could accelerate the algorithm computation.

Also, we are now developing a parallel genetic algorithm (see fig. 7) as a new heuristic to optimize λ value. The obtained results up today say us this is a very interesting research field to explore.

7. Acknowledgements

This work has been developed thanks to TRACER project [13] (TIC2002-04498-C05-01, Ministerio de Ciencia y Tecnologia, Spain, 2002-2005) working in it the following spanish universities: Universidad de Extremadura, Universidad de Malaga, Universidad Politecnica de Cataluna, Universidad de La Laguna and Universidad Carlos III de Madrid.

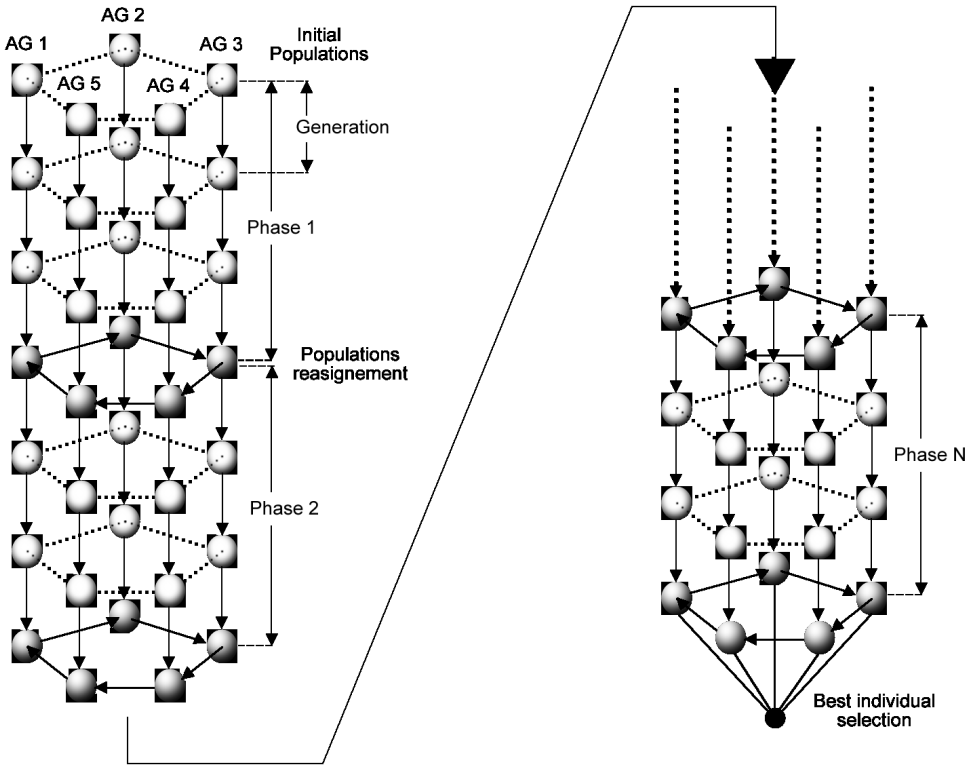

Figure 7. The scheme of the proposed parallel genetic algorithm

References

[1] T. Soderstrom et al.: System Identification. Prentice-Hall. 1989.
[2] L. Ljung: System Identification. Prentice-Hall. 1999.
[3] L. Ljung: System Identification Toolbox. The Math Works Inc. 1991.
[4] A Database for Identification of Systems: http://www.esat.kuleuven.ac.be/sista/daysi
[5] Royal Observatory of Belgium, Brussels: Sunspot Data Series. 2004.
[6] NOAA's National Geophysical Data Center: Sunspot numbers. 2004.
[7] J. A. Gomez, M.A. Vega, J.M. Sanchez: Parametric Identification of Solar Series based on an Adaptive Parallel Methodology. J. Astrophys. Astr. 26, pp 1-13. 2005.
[8] D. E. Goldberg: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley. 1989.
[9] I. Rechenberg: Evolutionsstrategie: Optimierung tech. sys. nach prinzipien der evolution. Formmann-Holzboog Verlag. 1973.
[10] S. Kirkpatrick, C. Gelatt., M. Vecchi: Optimization by Simulated Annealing. Science, 220, 4598, pp. 671-680. 1983.
[11] J.A. Gomez, J.M. Sanchez, M.A. Vega: Using Neural Networks in a Parallel Adaptive Algorithm for the System Identification Optimization. Lecture Notes on Computer Sciences 2687. Springer-Verlag, pp. 465-472. 2003.
[12] Gomez, J. et al.: Diseno de un coprocesador reconfigurable para Identificacion de Sistemas. II Jornadas sobre Computacion Reconfigurable. Univ. de Granada (Spain), pp. 221-226. 2002.
[13] TRACER: Tecnicas de Optimizacion Avanzadas para Problemas Complejos Estocasticos. http://tracer.lcc.uma.es. 2002.